Get In Step With StEPS: Standard Economic Processing System

Deborah Tasky, U.S. Bureau of the Census, Washington D.C.
Anne Linonis, U.S. Bureau of the Census, Washington D.C.
Scott Ankers, U.S. Bureau of the Census, Washington D.C.
Douglas Hallam, U.S. Bureau of the Census, Washington D.C.
Larry Altmayer, U.S. Bureau of the Census, Washington D.C.
Deborah Chew, U.S. Bureau of the Census, Washington D.C.

ABSTRACT

The Standard Economic Processing System, known as StEPS, is a generalized system being developed in the Economic Directorate of the U.S. Bureau of the Census to process over 100 current economic surveys. It is written entirely in SAS® and operates in a UNIX environment. This paper will describe what StEPS is and how it was designed to handle different surveys with different needs by using generalized programs and data structures. Modules within StEPS include many interactive SAS/AF® screens and use of SCL in a batch mode. Key modules within StEPS include Collection Activities, Review and Correction, Edit, Imputation, and Estimation.

INTRODUCTION

The U.S. Bureau of the Census has several directorates; the most widely known is the Decennial Directorate which conducts and processes the demographic decennial census. Another directorate, called the Economic Directorate, conducts economic censuses every 5 years and processes over 100 current surveys in the areas of retail, wholesale, service industries, manufacturing, and construction. These current surveys represent annual, quarterly and monthly programs. Examples include the Annual Retail Trade survey and the Annual Capital Expenditures survey.

Prior to 1995, each subject area had programming staffs devoted to the development of systems for their specific surveys. This resulted in 16 different processing systems, each performing similar functions. The separate staffs maintaining and managing these systems were often solving similar processing problems. Survey analysts were familiar only with the one processing system used to process their particular survey and not with the other processing systems being used.

In July 1995, the Economic Directorate formed a team to build a generalized processing system to process all of the current surveys. This team was made up of computer programmers, survey statisticians and mathematical statisticians. The team was charged with developing the system using SAS in a UNIX environment. This system was named the Standard Economic Processing System (StEPS). The primary goals of the StEPS were to:

! Reduce resources required for system maintenance and for survey migration to other platforms.

! Standardize survey procedures used in data analysis and management.

! Improve timeliness for new surveys by eliminating analyst retraining and the development of custom survey processing software.

! Provide greater staffing flexibility for analysts and programmers to process different surveys by providing a processing system common to all surveys.

! Make all survey data available to all users (with certain security restrictions).

! Provide a common structure to make it easier to implement improvements for ALL surveys in the system.

! Shift more control to the survey analysts by allowing them to set survey processing parameters and run processes themselves.

Three surveys were processed using an early version of StEPS in 1998. In 1999, 50 annual surveys are using StEPS to process data for the 1998 statistical survey year.

HOW WAS StEPS DEVELOPED?

The team devoted their first year to gathering user requirements and becoming proficient with the SAS6.12 features of SAS/AF and SCL. They held focus groups with the survey analysts, examined the functionality of the existing 16 processing systems, and developed interactive screen prototypes.

Although it was clear that the surveys required similar modules (i.e., data capture, review and correction, edit, imputation, estimation), it was also evident that some surveys had specific requirements that were not needed by other surveys. For example, most surveys need edits to determine if an item from the form is reported, but the exact items to test for are dependent on the specific survey collection form. Also, basic techniques are required in the imputation module; but when to use which technique, on which item, and with what contributing data are dependent on the specific survey.

As a result, the major challenge for the StEPS Development Team was to develop a processing system with generalized code that could:

! Run on any survey and statistical period of data.

! Process items from one survey that are completely different than items from another survey.

! Accommodate surveys that change their data collection forms and/or collect different items from one statistical period to the next.

! Accommodate surveys that may have a different set of respondents from one statistical period to the next.

! Allow for a survey to 'customize' its set of processing requirements.

HOW IS StEPS GENERALIZED?

To generalize the StEPS, the team decided on four major design concepts:

1) Design a set of standard data structures that remain the same, regardless of the survey and its data.

2) Use parameters (stored in general data structures) to drive the survey-specific processing requirements.

3) Generate a 'fat' record data set on the fly for certain modules (which contains the needed information for that process), perform the process, and then determine and apply updates to the original files.

4) Standardize field names and possible values for similar concepts.

Each of these design concepts is described in more detail below.

Standard Data Structures

The data collected for the Economic area surveys encompasses a wide range of information. For example, data might include the number of square yards of cotton textiles, the annual sales of a computer store, the Research and Development expenditures for a large company, and/or the amount of fuel used by another. Each of these pieces of data is similar in that it is numeric, and StEPS can therefore store the data in the same structure. Information for each of these data items is contained in a dictionary data set that tells the system what each item means. These data set structures are identical for every survey processed in StEPS.

Because StEPS processes many different types of surveys, it was necessary to allow for different statistical processing periods of data - a concept known as a "stat period". For example, in early 1999, data were being collected for 1998 surveys. The stat period for processing these surveys is therefore '1998'. At the same time, some surveys were still producing estimates for the 1997 surveys. The stat period for these surveys is '1997'. The stat period concept can become complicated when processing monthly and quarterly surveys, since the data collection period and the actual processing period for the data can overlap. StEPS must be able to treat any stat period as the "base" (or current) stat period and any stat period other than the "base" as it relates to the base.

Libnames Used in StEPS: The code for StEPS uses standard libnames. When a user chooses a survey, a libname called SURVLIB is set up. The physical location or directory for this libname is stored in a data set called CENTRAL.SURVEYS:

CENTRAL.SURVEYS data set:
(select variables)

1 SURVEY	Char	Survey identifier
2 SURVNME	Char	Survey name
...		
6 SURVDIR	Char	Directory of top-level survey info: SURVLIB
...		

Once a user selects a survey and SURVLIB is set up, a data set named SURVLIB.VSTATPS is opened. The user can then select the stat period of data to access, from a list of stat periods available for that particular survey.

SURVLIB.VSTATPS data set
(select variables)

1 SURVEY	Char	Survey identifier
2 SURVNME	Char	Survey name
3 STATP	Char	Statistical period
4 STATUS	Char	Status A: Active Status N: Not active
5 DATADIR	Char	Data directory: DATALIB
6 PARMDIR	Char	Parameter directory: PARMLIB
7 DATASDIR	Char	Stat period specific data: DATA##
8 PARMSDIR	Char	Stat period specific parameters: PARM##
9 ARCHDIR	Char	Stat period archive directory: ARCH##
10SPRGDIR	Char	Directory for survey-specific programs: SPRGLIB
...		

Non-stat period related libnames: DATALIB and PARMLIB are then set up. The base stat period is assigned the following libnames: DATA00 and PARM00. By having all the physical directories stored in data sets, programmers have control over where the data is physically located. There are two routines that are used to set up the libnames for StEPS. One is interactive and written in SCL. It is called when the interactive part of StEPS is invoked. The second routine is a macro which is called in batch, and in non-interactive StEPS routines. These routines also control which libnames are invoked with SAS/Share servers and which ones are not. The following illustrates how the macro is invoked:

```
%setlibs(survey=MA22Q,statp00=1998a1,
statp01=1997a1, statp02=1996a1);
```

This design sets up standard libnames that simply point to different physical locations based on what is stored in these data sets, regardless of what survey or stat period is used.

Data Set Structures: Because standard data sets are used in StEPS and because new data sets for new stat periods and/or surveys are constantly being created, it was necessary to design a way to easily copy data structures into the survey directories. A separate central library called DSDEF ("data set definition") was created. Empty data set structures are stored within this library. When new data sets are needed, these empty data sets are used as the shell for the new data set, ensuring that the structure is correct. A catalog of proc sql code creates the empty structure. When the structure must be changed, the code is changed in the catalog, the DSDEF data structure is recreated, and the new data structure propogated to the survey data sets. DSDEF is also used to ensure that existing data sets have not been corrupted in some way.

Major StEPS Data Sets: Major data sets in the StEPS system are the <u>control</u> files and the <u>item</u> files. The control files contain name and address information for each respondent (called the ID) along with processing information for a particular statistical period including sampling, mailing, collection, and check-in information. The item files contain numeric information for each ID, either from the form itself, from other sources, or derived as a function of other items. These files are often referred to as the 'skinny' files because there is a separate record for each ID/item. The actual content of both the control and item files is driven by the data dictionaries, which are described in more detail below.

Survey Parameters

The survey parameters in StEPS are referred to as "survey specifications". Two of the most important survey specification files in StEPS are the data dictionaries for the item and control files. The item data dictionary contains a record for each item in the survey, along with various processing fields to indicate whether the item is correctable, is to be weighted, etc. The control file data dictionary contains a record for each of the standard variables (that the team determined each survey must have), as well as a record for each control-type variable specific to a survey. These data dictionaries are important in the generation of the 'fat' record (described in the next section) which is used in various modules within StEPS.

StEPS has many other survey specification files which define the processing rules for a particular survey. Users define rules for editing data, formulas for creating derived items, imputation methods, 'where clauses' to select appropriate cases to mail, and table specifications for estimation.

For many of the modules, standard programs read the survey specification files and generate survey-specific code. For other modules, the survey specification files are simply read and used in a standard program (e.g., storing the information in SCL lists).

'Fat' Record and Updates to Original Files

As discussed above, the generalized nature of StEPS requires that item data be stored in a flexible structure. The StEPS "skinny record" design gives us this flexibility and allows surveys to easily adapt to StEPS. Essentially, the "skinny" design dictates that each observation of an item data set contain all relevant information about an item, (i.e., reported value, edited value, weighted value, flags). Considering the variety of surveys that StEPS will process and the varying number of items that each survey possesses, this design structure works well. However, some StEPS processing is more easily accomplished if all data relating to an ID are stored in one observation. The CNVT macro was developed for this purpose.

The CNVT macro builds a SAS data set containing observations by ID. Each observation includes item and control file information related to that ID. The program's default behavior is to include in its resulting data set all items and control file data for all valid statistical periods defined for a given survey. This can potentially result in a data set with extremely large numbers of variables in each observation. For this reason, this data set is commonly called the "fat record" data set.

Because the output data set can become very large and may include unnecessary data for a particular task, flexibility was added to the program to allow the calling program to customize the results to suit its needs. This flexibility is facilitated through the use of macro keyword parameters. The program presently contains nine different keyword parameters that can significantly pare down the size of the output data set. The calling program can dictate which items it wants, the type of items to include (for example, only include “reported” data), and which statistical periods to include. Also, the number of observations written out can be restricted by providing the program a finite list of IDs or a conditional statement that is used in a subsetting “if” statement.

Once CNVT has created the desired “fat record” data set, StEPS modules such as edit and imputation can then operate on one data set.

Since the fat data set is processed and updated by many different survey processing modules, it was necessary to design a general way to take the updates from the fat data set and apply them back to the standard StEPS data structures. This was done by saving an original copy of the fat data set and then comparing it to the modified fat data set. Based on which module is being processed, certain fields and flags are compared and updates written back to the item data set.

The team’s first attempt at this was to use the PROC COMPARE. This was found to be cumbersome for this process. As an alternative, the team wrote an SCL program without a frame, which can be run both interactively and in batch. The SCL program gets the number of variables from the fat record using the ATTRN function and loops through the variables with the VARNUM and VARNAME functions. Because StEPS has a standard way of naming the variables in the fat record, the program knows which variables to compare to the original data set. If there has been a change, an update is written to the appropriate item data set. A record is usually written to the audit trail (DATALIB.ITAUDIT) as well, and other information is written to the log. For something this complicated, the control that SCL provided was critical to effectively perform this function. Different modules check different variables and flags. This routine is general and works for every survey, even though each fat record for each survey is very different.

Standard Fields and Values

During the first year when the team was gathering requirements, they learned that many systems had similar processing concepts, but often different field names and different values for these concepts. The team worked with the subject areas to determine a set of standard field names and values. For example, to determine whether an ID is still valid and should be tabulated, a field called STATUS was created with a value of ‘A’ indicating that the ID is active and a value of I’ indicating that the ID is inactive. This was not an easy task and for some seemingly simple fields, the resolution took many months.

Standardizing these fields and values greatly contributed to the team’s ability to develop the generalized system. Programs to determine which cases to mail, and which cases to impute and tabulate could all be driven from standard fields which every survey was using.

OTHER PROGRAMMING INFORMATION

SAS Share

Because there are many users accessing the same files, it is necessary to use SAS/Share to control multi-user access to the files. One central share server was created for StEPS files and programs used by all surveys. A separate server was then created for each survey processed in StEPS. The libname set-up routines that are used invoke these libraries with the servers.

StEPS contains a program that loops through CENTRAL.SURVEYS and each survey’s SURVLIB.VSTATPS data sets and generates the code that points the directories to the appropriate SAS/Share server.

Listed below is sample code used for creating one survey’s SAS/Share server process:

```
* ;
* ;
* set up sas share for survey TEST ;
options comamid=tcp;
libname _all_ clear;
libname shr1 '/steps/test' ;
libname shr2 '/steps/test/d1993a1' ;
libname shr3 '/steps/test/d1994a1' ;
libname shr4 '/steps/test/d1995a1' ;
libname shr5 '/steps/test/d1996a1' ;
libname shr6 '/steps/test/d1997a1' ;
libname shr7 '/steps/test/d1998a1' ;
libname shr8 '/steps/test/data' ;
libname shr9 '/steps/test/p1993a1' ;
libname shr10 '/steps/test/p1994a1' ;
libname shr11 '/steps/test/p1995a1' ;
libname shr12 '/steps/test/p1996a1' ;
libname shr13 '/steps/test/p1997a1' ;
libname shr14 '/steps/test/p1998a1' ;
libname shr15 '/steps/test/parms' ;
libname shr16 '/steps/test/programs';
proc server id = TEST
  alloc
  log=all;
run;
```

Any libname statements that involve the directories above will include "server=TEST" and will use SAS/Share. This is done automatically for most users with the SETLIBS macro and from within the interactive part of the system.

Programming Standards

The StEPS is designed for minimal system maintenance. As such, programmers document new code as they develop it. As an evolving system with continual updating, documenting changes makes it easier to make program modifications, as the need for them arises. This documentation includes program name, author, date of completion, and description of program or modification. Programmers also adhere to several common rules designed for readability, especially given the code-generating nature of StEPS. This ensures readability for both static and generated code.

The system uses SAS programming features designed for more efficient processing with less coding. For example, upon entering StEPS, an autoexec.sas file automatically invokes an autocall library containing several macros called regularly during the implementation of StEPS. For file management, most SAS source code itself, as well as frame, scl, pmenu and other types of files, are stored in SAS catalogs.

Since much of the code for StEPS uses SCL for either batch or FRAME execution, there are standards for developing SCL code. StEPS uses the DATALISTC or DATALISTN functions to display selection list windows which contain values from SAS data sets. Before calling either function, the window is placed using the standard 'call wregion(10,10,...)' statement. Since the SAS source code programs are stored in catalogs, they can be run from an SCL program by copying it into the preview buffer and then submitting it using a submit block.

Since there are several parameters used in whichever module a person might be in at a given moment, there are several global SAS macro variables. Examples are SURVEY, STATP00 (base statistical period), USRNME (username last changing an element), and DATAPRIV (privilege to change data). Additionally, since UNIX directory storage locations are changed from time to time because of system management, environment variables are used. An example of a UNIX environment variable is CENTRAL (the central StEPS directory), which can be used in a SAS FILENAME statement, when preceded by a '$' ($CENTRAL).

Another feature common to all of the screens in StEPS is function key assignment. Many keyboard keys have standard definitions in the SAS system, as are shown in the keys window. There are a few keys which are universal for the entire